Europäisches Patentamt | European Patent Office | Office européen des brevets

EP00/01929

Bescheinigung | Certificate | Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. | Patent application No. | Demande de brevet n°

99250083.5

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

I.L.C. HATTEN-HECKMAN

DEN HAAG, DEN
THE HAGUE,
LA HAYE, LE 17/05/00

Europäisches Patentamt
European Patent Office
Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.:
Demande n°: 99250083.5

Anmeldetag:
Date of filing:
Date de dépôt: 19/03/99

Anmelder:
Applicant(s):
Demandeur(s):
DEUTSCHE THOMSON-BRANDT GMBH
78048 Villingen-Schwenningen
GERMANY

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
Method and apparatus for implementing trickplay modes in a data stream recorder

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
/

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du dépôt:

Bemerkungen:
Remarks:
Remarques:

THOMSON multimedia
CORPORATE RESEARCH
HANNOVER

Page 1	Date	00.00.00
	Index	Entry Points
of 8 pages	Name	Marco WINTER

THOMSON multimedia CONFIDENTIAL

PD990019

1

TRICKPLAY SUPPORT PROPOSAL

~~An "I-picture flag" mechanism for trickplay support has been proposed by MEI~~ in document TG1-3-11/24/1988-9C. Also, LGE ~~has proposed~~ a mechanism for improving addressing ~~accuracy.~~ As a proposal for unification and compromise, ~~THO has considered these concepts and now proposes~~ the following trickplay concept is proposed for a DVD stream recorder. (In the invention)
This proposal describes a way to implement Entry Points (EP). The resulting EPs have a resolution range from "2 SOBUs" up to "application packet" exact. The precision depends on the used DVD Streamer, i.e. whether the DVD Streamer knows the application and how much RAM memory is available and so on. With other words: the precision is manufacture matter.

Principle The invention is applicable in a corresponding way for other types of data stream recorders.

Each SOB contains its own EP information (EPI). This EPI consists of a general information, one coarse list and a fine list.

The coarse list is called the Entry Point Start Map (EPSM). The EPSM consists of N flags (N is the number of SOBUs of this SOB). Each flag belongs to one SOBU. The flag indicates

- into the corresponding SOBU or into the next SOBU points an EP
- no corresponding EP exists to this flag

The second list is called the Entry Point Location List (EPLL). The EPLL contains the application packet exact locations of all EPs. For each EP indicating EPSM flag exists one location information inside EPLL. This location information contains either

- the sector & application packet location of the start of the EP and
- the sector & application packet location of the end of the data which starts at the EP (e.g. the end of the I-frame) and
- the PTS of the EP

or

- the start APAT of the EP
- the end APAT of the random access unit (e.g. the end of the I-frame) and
- the PTS of the EP

The part inside EPLL which contains the start location is called the Entry Point Start Location List (EPSLL).

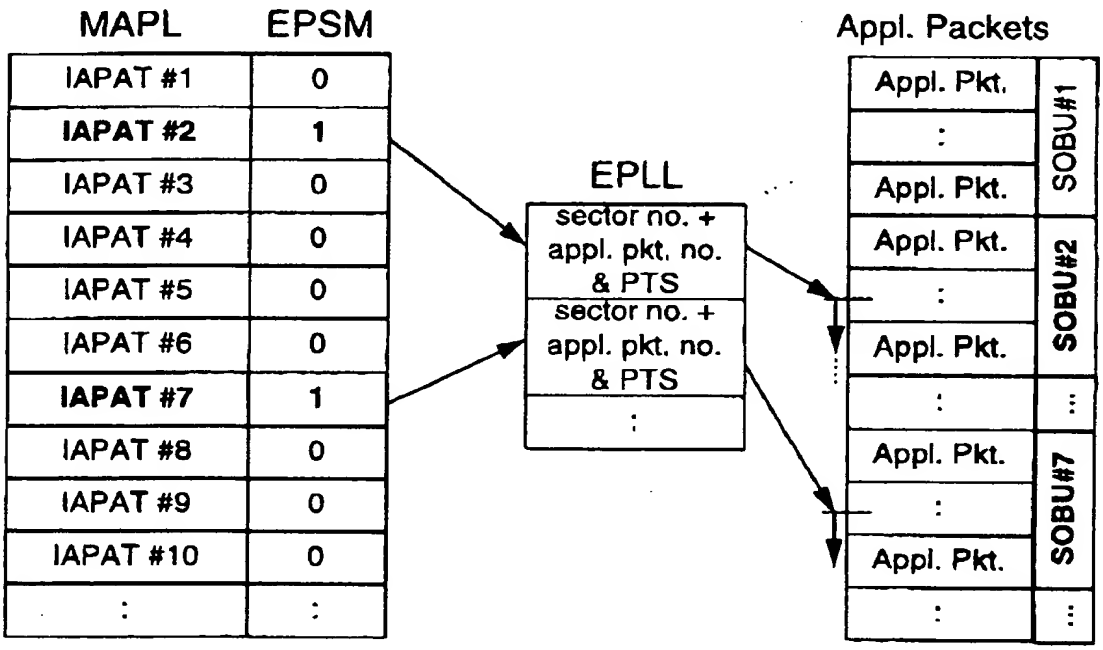
THOMSON multimedia CORPORATE RESEARCH HANNOVER	Page 2 of 8 pages	Date	00.00.00
		Index	Entry Points
		Name	Marco WINTER

2

The part inside EPLL which contains the end location is called the Entry Point End Location List (EPELL).
The trickplay (e.g. fast forward) will be performed by selecting the desired EPs (e.g. each second EP) via EPSM.

The generating of EPSM, EPSLL and EPELL during SOB recording is optional (manufacture matter). The use of EPSM, EPSLL and EPELL for trickplay is also optional (manufacture matter). It's mandatory to update the EPSM and EPLL in the case of editing.

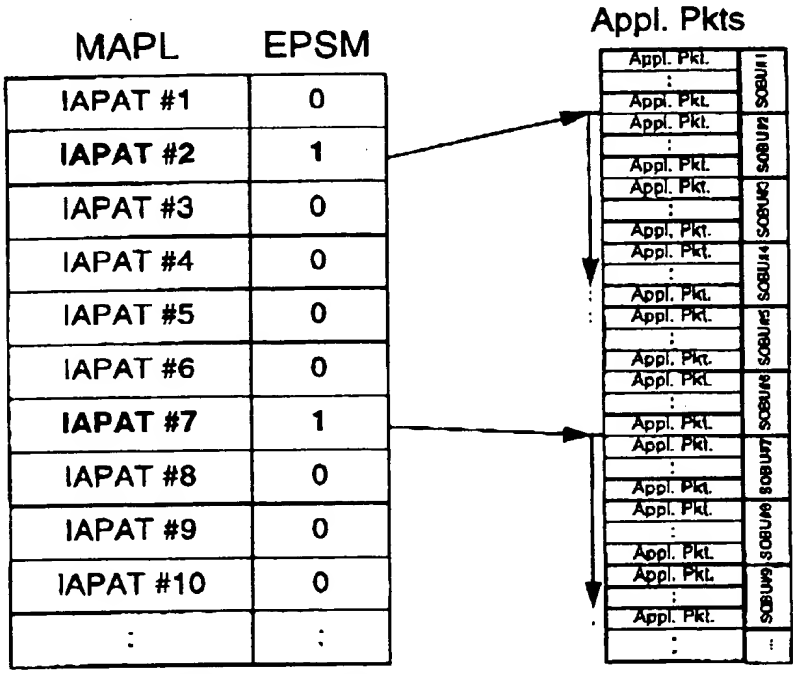
Three examples:



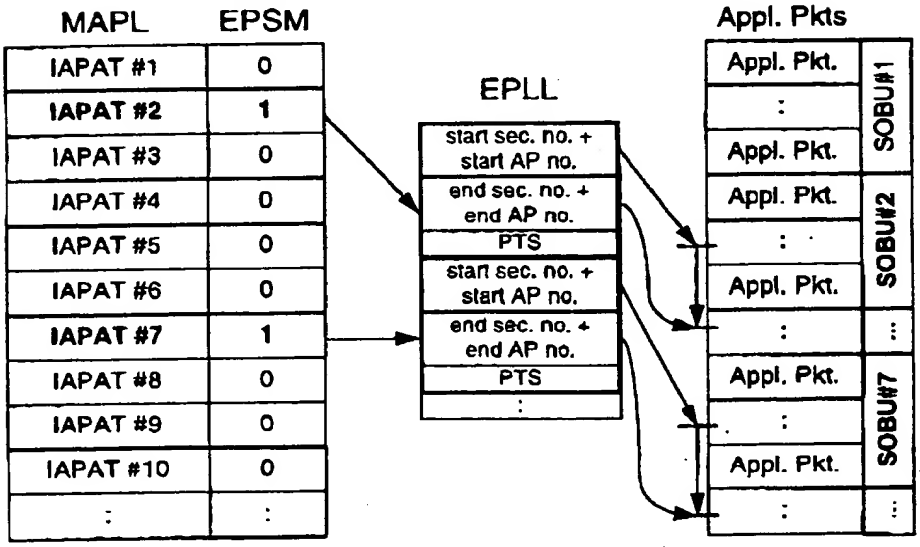
Access via EPSM and EPLL

THOMSON multimedia
CORPORATE RESEARCH
HANNOVER

3



Access via EPSM, but without EPLL



EPLL also contains end of EP information

4

Generating of EPSM and EPLL

The DVD Streamer specification shall define the syntax of the EPs, not the generating or using of the EPs. However, here are some possible example methods to generate EPSM and EPLL:

- **Method 1:** the application sends after transmission of the stream special data which contains a list of EPs as APATs, i.e. each APAT of the list is the APAT of one of the just recorded application packets. The streamer must assign each APAT to the corresponding application packet:
  - a "high end" streamer did generate a special list during stream recording. This list contains the APAT values of each recorded application packet and the corresponding location in the stream (sector no. and application packet no.). When the application sends the EP list as a list of APATs, then the streamer is able to generate both lists: EPSM and EPLL.
  - a "simple" streamer has not enough memory to generate a list with APATs and application packet location information inside the local RAM. Therefore, the streamer will generate only the EPSM, but not the EPLL. Later, a "high end" streamer could generate the exact EPLL (e.g. during idle-mode of the "high end" streamer)
- **Method 2**: the streamer contains dedicated hardware to parse the incoming stream, i.e. the application is known by the streamer. This parser recognizes automatically entry points like I-pictures. It's simple with such an additional hardware to generate EPSM and EPLL during stream recording.
- **Method 3**: the application uses special digital interface commands to mark an application packet as EP during transmission of the stream to the streamer. So, the streamer is able to generate EPSM and EPLL parallel during stream recording. This needs special definitions for the digital interface specification.
- **Method 4**: the application doesn't know anything about the streamer. In this case EPs won't be generated. Later, a "high end" streamer could generate the missing EPSM and EPLL (e.g. during idle-mode of the streamer).

Apply Trickplay

Without end of EP information

The trickplay (e.g. fast forward) will be performed by searching for the desired EPs (e.g. each second EP) inside the EPSM. With the EPLL (if exists) the exact location of the first application packet of the EP is known. Without the EPLL the streamer assumes, that the EP is located anywhere in the SOBU indicated by

the EPSM or in the following SOBU. The streamer jumps to this position and starts the transmission of the application packets to the application. The streamer stops the transmission after transmission of a fixed size of data (e.g. 1.8 Mbit or till the next EP) and jumps to the next desired EP. If the streamer knows the application, then it could parse the stream during transmission of the EP and would stop the transmission, when the end of the EP is reached (e.g. end of I-picture).

With end of EP information

The only different to the method "Without end of EP information": if the EPLL exists, then the transmission of an EP to the application stops with the transmission of the last application packet of the EP.

2.2.3.3 Stream Object Information (SOBI)

As shown in Figure 2-6, a Stream Object information (SOBI) consists of the Stream Object Information General Information (SOBI_GI), the Mapping List (MAPL) and the Entry Point Data (EPD), if any.

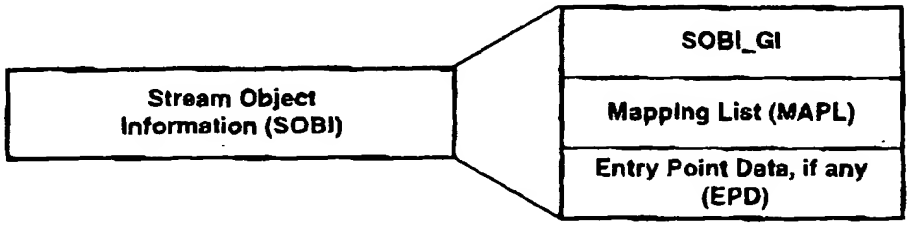


Figure 2-6: Structure of a Stream Object Information

2.2.3.3.1 Stream Object Information General Information (SOBI_GI)

	Contents	Number of Bytes
(1) SOB_TY	SOB Type	1
(2) SOB_REC_TM	SOB Recording Time	5
(3) SOB_STI_N	SOB Stream Information Number	1
(4) EP_FLAGS	Entry Point properties	1
(5) SOB_S_APAT	SOB Start APAT	6
(6) SOB_E_APAT	SOB End APAT	6
(7) SOB_S_SOBU	first SOBU of this SOB	4
(8) IAPAT_SUM_INIT	initial value for IAPAT summation	4
(9) MAPL_ENT_Ns	number of Mapping List entries	4
	Total	32

(1) SOB_TY
Describes the Stream Object Type, containing bits for Temporal Erase state (TBD) and for Copy Generation Management System (TBD).

(2) SOB_REC_TM
Describes the recording time of the associated Stream Object in DVD Stream Recording's Date and Time Describing Format defined above.

(3) SOB_STI_N
Describes the index of the BOB_STI which is valid for this Stream Object.

(4) EP_FLAGS
Indicates whether Entry Point Data of this SOB exists and whether the stream itselves contains Entry Point information. If Entry Point Data exists, then EP_FLAGS also describes several properties of the Entry Points. The Entry Point Data consists of the number of Entry Points and the lists EPSM, EPSLL, EPELL and PTSLL. The content of EP_FLAGS:

b7	b6	b5	b4	b3	b2	b1	b0
EP marks in stream	EP data exist	Time Based	EPSLL exists	EPELL exists	PTSL exists	reserved	

EP marks in stream ... 0b: no Entry Point marks inside the stream of this SOB

1b: there may be Entry Point marks inside the stream of this SOB. This state is even allowed, when no Entry Point data of this SOB exists.

EPD exists ... 0b: no Entry Point Data exists for this SOB. The bits b5, b4, b3 and b2 of EP_FLAGS shall be set to 0.
1b: there is Entry Point Data of this SOB (behind the MAPL)

Time Based ... 0b: EPSLL (if exists), EPELL (if exists) and EPSM consists of sector based location information
1b: EPSLL (if exists), EPELL (if exists) and EPSM consists of time based location information

EPSLL exists ... 0b: no EPSLL of this SOB exists
1b: EPSLL of this SOB exists

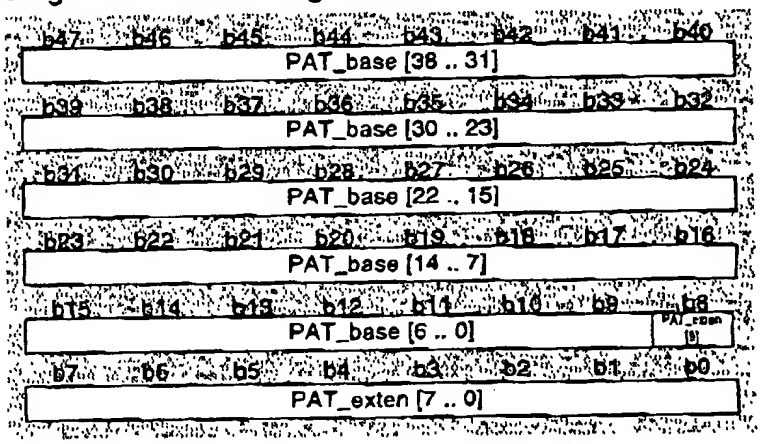
EPELL exists ... 0b: no EPELL of this SOB exists
1b: EPELL of this SOB exists

PTSL exists ... 0b: no PTSL of this SOB exists
1b: PTSL of this SOB exists

(5) SOB_S_APAT

Describes the start Application Packet Arrival Time of the Stream Object, i.e. the packet arrival time of the first packet belonging to the SOB. SOB_S_APAT is described in DVD Stream Recording's PAT Describing Format defined in the following.
PATs are divided into two parts, namely a base part and an extension part. The base part holds the so-called 90kHz unit value, and the extension part holds the less significant value measured in 27MHz.

DVD Stream Recording's PAT Describing Format:



PAT_base ... PAT's base value measured by 90kHz unit
PAT_exten ... PAT's extension value measured by 27MHz unit ($0 \le$ PAT_exten < 300)

PAT in seconds = PAT_base / 90,000 Hz + PAT_exten / 27,000,000 Hz

Note: For a unique representation of times, PAT_exten must be in the range of $0 \le$ PAT_exten < 300. Together, PAT_base and PAT_exten cover a range of more than 1696 hours.

(6) SOB_E_APAT
Describes the end Application Packet Arrival Time of the Stream Object, i.e. the packet arrival time of the last packet belonging to the SOB, in DVD Stream Recording's PAT Describing Format.

(7) SOB_S_SOBU
Describes the number of the start Stream Object Unit, i.e. the Stream Object Unit containing the first Application Packet of the Stream Object.

(8) IAPAT_SUM_INIT
Describes the initialization value which will be added to any summation of IAPAT entries during Mapping List usage. This means that the summation of the first i IAPATs will result as

$$SUM\_IAPAT(i) = IAPAT\_SUM\_INIT + \sum_{k=1}^{i} IAPAT(k) \qquad \text{(eq 2-1)}$$

for i = 1,2,...,MAPL_ENT_Ns
IAPAT_SUM_INIT is especially defined to handle the case when the start of a SOB is permanently deleted. In such a case IAPAT_SUM_INIT is used for convenience to keep all the APATs related to this SOB unchanged.

At the time of original recording, IAPAT_SUM_INIT shall be chosen as

$$IAPAT\_SUM\_INIT = \left\lceil \frac{APAT_{firstPacket}}{2^{MTU\_SHFT}} \right\rceil \qquad \text{(eq 2-2)}$$

where $APAT_{firstPacket}$ shall denote the Application Packet Arrival Time of the first Application Packet of the Stream Object. If $APAT_{firstPacket}$ is chosen sufficiently small (e.g. 0), IAPAT_SUM_INIT will result as 0.

When the start of the SOB is deleted, IAPAT_SUM_INIT will be simply incremented by the IAPATs of the erased SOBUs (see Annex).

Note: $\lceil \bullet \rceil$ *is the ceiling function, see Section 0.3.1.*

(9) MAPL_ENT_Ns
Describes the number of Mapping List entries to follow after SOBI_GI.

9

2.2.3.3.3 *Entry Point Data*

As shown in Figure 2-9, the Entry Point Data (EPD), if any, consists of the Entry Point General Information (EP_GI), the Entry Point Start Location List (EPSLL), the Entry Point End Location List (EPELL) and the Presentation Timestamp List (PTSL). Which parts of the EPD exists is indicated by EP_FLAGS (defined in the SOBI_GI).

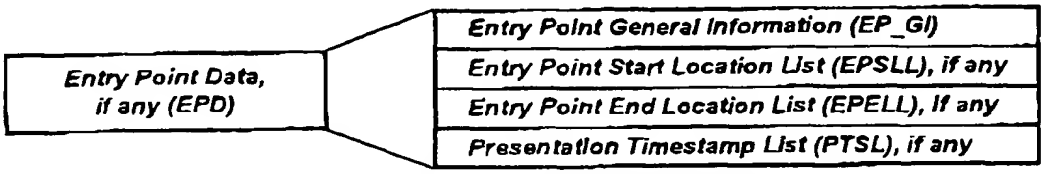


Figure 2-9: Structure of the Entry Point Data

2.2.3.3.3.1 Entry Point General Information (EP_GI)

EP_GI only exist, when EP_FLAGS of SOBI_GI indicates that Entry Points Data exists.

	Contents	Number of Bytes
(1) EP_Ns	number of Entry Points	4
(2) EPSM	Entry Point Start Map (MAPL_ENT_Ns entries)	(MAPL_ENT_Ns*SM_SZ+7) div 8
	Total	4+(MAPL_ENT_Ns*SM_SZ+7) div 8

(1) EP_Ns

Describes the number of Entry Points, indicated by the EPSM.

(2) EPSM

The Entry Point Start Map indicates the SOBUs of the SOB containing Entry Points, i.e. for each SOBU exists exactly one EPSM entry. So, the EPSM consists of MAPL_ENT_Ns entries. Each EPSM entry indicates the number of accessable Entry Points of the corresponding SOBU. Exact EP_Ns Entry Points are indicated by the EPSM. EPSM shall be byte aligned. If the concatenated EPSM entries consists of a number of bits which is not an integer multiple of 8, then the LSBs of the last byte of the EPSM shall be the necessary additional padding bits. These alignment bits shall be set to 0. Figure 2-10 shows an example of an EPSM and its corresponding SOBUs, ('EPSM resolution' = 00b).

The precision of the EPSM depends on the settings in EP_FLAGS:

- *"Time Based" is 1b, no EPSLL exists and no EPELL exists: an Entry Point indicated by EPSM is located in the corresponding SOBU or in the following SOBU.*
- *all other cases: an Entry Point indicated by EPSM is located in the corresponding SOBU of the SOB.*

The maximum number of addressable Entry Points per SOBU (SM_SZ) is defined by the 'EPSM resolution' flag inside EP_FLAGS

EPSM resolution	00b	01b	10b	11b
SM_SZ (max number of EPs per SOBU)	1	3	15	255

Figure 2-10b shows an EPSM example with 'EPSM resolution' = 01b.

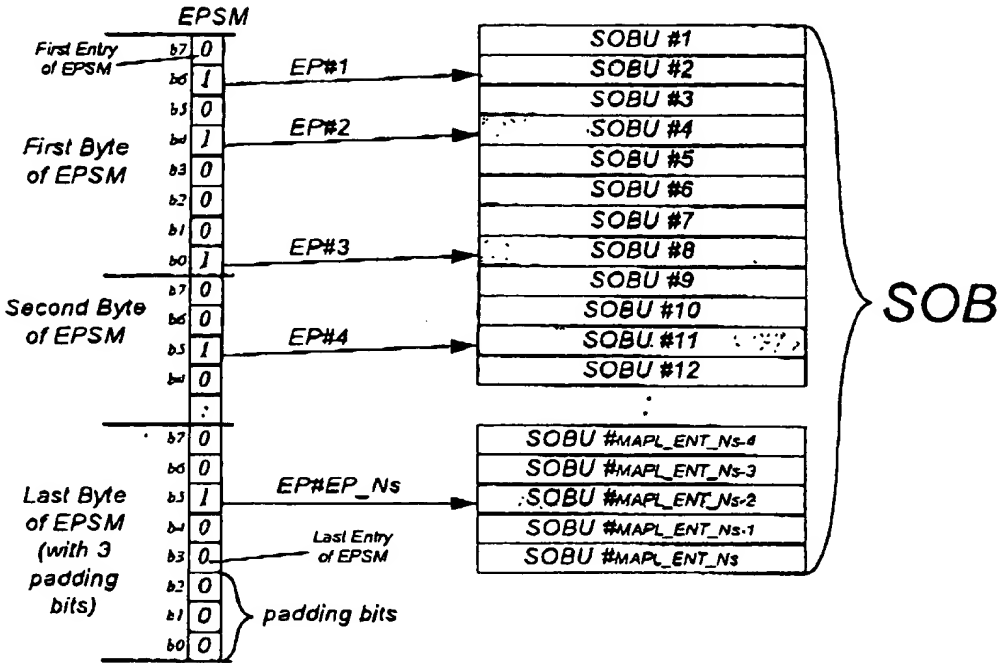


Figure 2-10: Example of an EPSM and its corresponding SOBUs

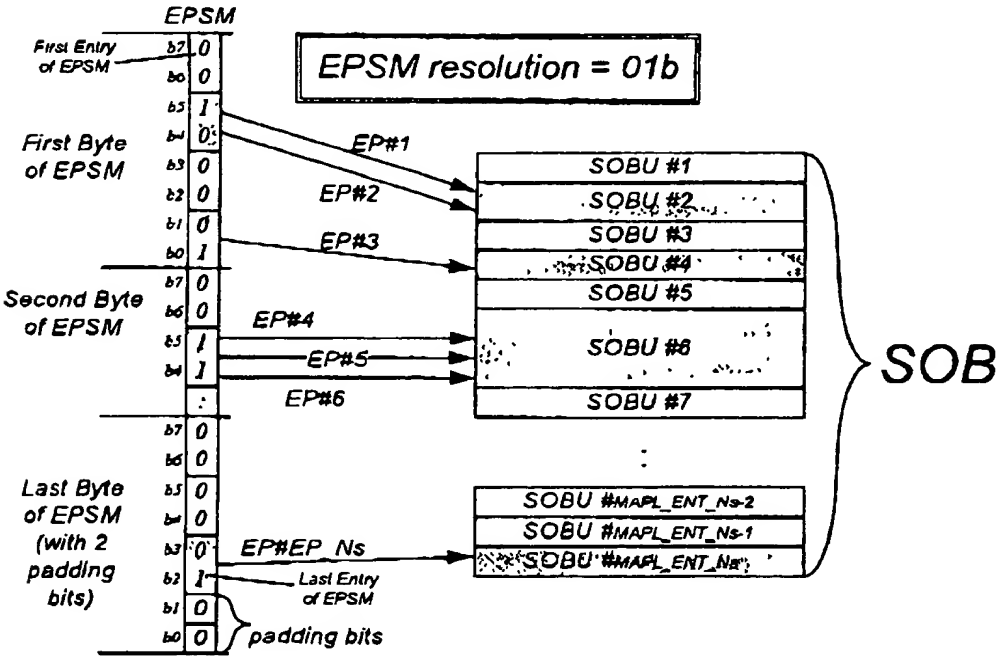


Figure 2-10b: Example of an EPSM with 'EPSM resolution' = 01b

The structures of the EPSM with 'EPSM resolution' = 10b and 11b are appropriate.

2.2.3.3.3.2 Entry Point Start and End Location List (EPSLL and EPELL)

EPSLL is a list of location information to find the application packet where the random access units start, i.e. if EPSLL exists, then each Entry Point of its access unit has exactly one corresponding EPSLL entry.

EPELL is a list of location information to find the application packet where the random access units stop, i.e. if EPELL exists, then each Entry Point of its access unit has exactly

one corresponding EPELL entry. Each application packet, indicated by the EPELL entries, is the last application packet belonging to the random access unit.
The entries of EPSLL and EPELL are in ascending order, i.e.

- *the first EPSLL entry and the first EPELL entry belong to the first occuring Entry Point inside EPSM*
- *the second EPSLL entry and the second EPELL entry belong to the second occuring Entry Point inside EPSM*
- ...
- *the last EPSLL entry and the last EPELL entry belong to the last occuring Entry Point inside EPSM*

The entries of EPSLL and EPELL of the same SOB are always in the same format. It's either the time based format (Time Based flag of EP_FLAGS is set to 1b) for both tables or the sector based format (Time Based flag of EP_FLAGS is set to 0b) for both tables.

2.2.3.3.3.2.1 Sector Based EPSLL and EPELL Entries

If the "Time Based" flag of EP_FLAGS is set to 0b, then the entries of EPSLL and EPELL are sector based, i.e. their entries are defined as

	Contents	Number of Bytes
(1) SOFF	sector offset	1
(2) AP_NO	Application Packet number inside the sector	1
	Total	2

(1) SOFF
EPSLL: SOFF is part of the location information. The sector #SOFF is the sector offset inside the corresponding SOBU of this EPSLL entry, i.e. SOFF = 0, 1, ..., SOBU_SZ-1. SOFF=0 means the first sector of the SOBU.
EPELL: SOFF is part of the location information. The sector #SOFF is the sector offset to the first sector of the corresponding SOBU of this EPELL entry, i.e. SOFF = 0, 1, ..., 255. The indicated sector may be outside of the corresponding SOBU of this EPELL entry (see first entry in EPELL of Figure 2-11). SOFF=0 means the first sector of the SOBU. If EPSLL exists, then SOFF shall be greater than or equal to the corresponding SOFF of EPSLL.

(2) AP_NO
AP_NO is part of the location information. Let N be the number of application packet starts in sector #SOFF. Then, AP_NO indicates one of these N application packets, i.e. AP_NO = 1, 2, ... , N. AP_NO=1 means the first starting application packet of the sector.

Figure 2-11 shows on the left side a sector based example of EPSLL and EPELL.

2.2.3.3.3.2.2 Time Based EPSLL and EPELL Entries

If the "Time Based" flag of EP_FLAGS is set to 1b, then the entries of EPSLL and EPELL are time based, i.e. their entries are defined as

	Contents	Number of Bytes
(1) EP_APAT	APAT of an Application Packet	6
	Total	6

(1) EP_APAT
EPSLL: EP_APAT is the location information. EP_APAT is the APAT of an application packet inside the corresponding SOBU of this entry.

EPELL: EP_APAT is the location information. EP_APAT is the APAT of an application packet inside the corresponding SOBU of this entry or in one of the following SOBUs. If EPSLL of the same SOB also exists, then the EP_APATs of EPELL shall be always greater than or equal to the corresponding EP_APATs of the EPSLL.

Figure 2-11 shows on the right side a time based example of EPSLL and EPELL.

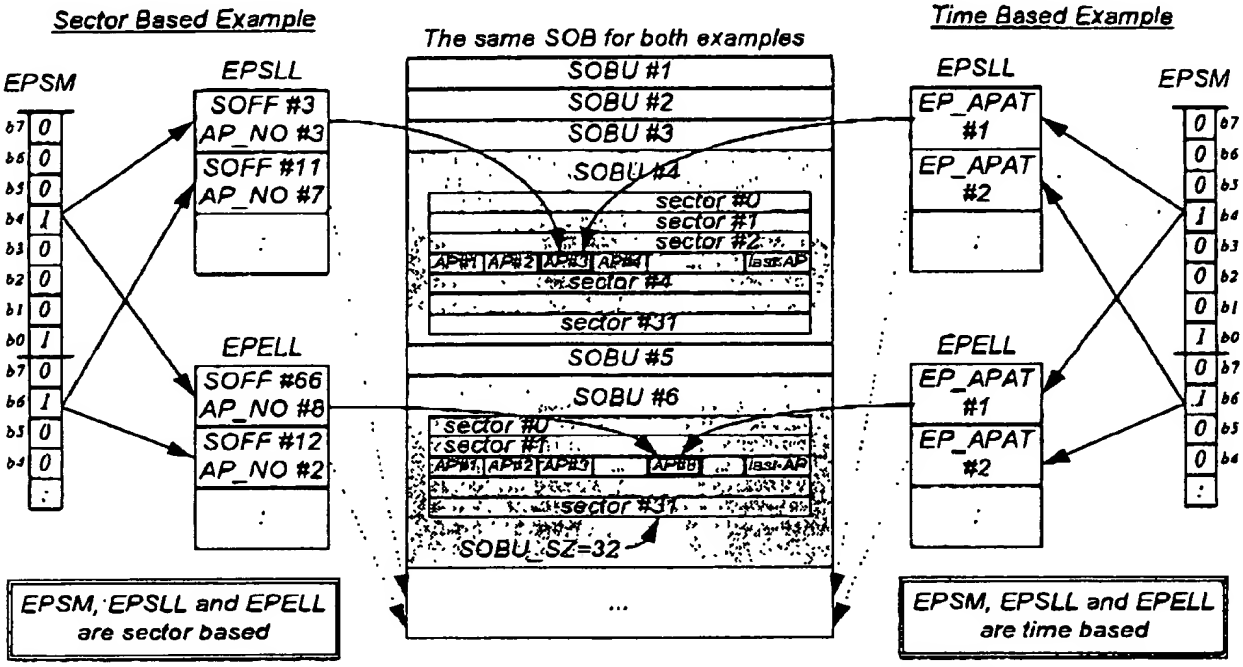


Figure 2-11: Two EPSM, EPSLL and EPELL examples in one figure: on the left side sector based, on the right side time based. Note: either sector based or time based is allowed for a SOB

2.2.3.3.3.3 Presentation Timestamp List of the Entry Point (PTSL)

PTSL is the list of the Presentation Timestamps of all the Entry Points of this SOB, i.e. if PTSL exists, then each Entry Point has exactly one corresponding PTSL entry. So, PTSL has either 0 or EP_Ns entries.

The entries of PTSL are in ascending order, i.e.

- *the first PTSL entry belongs to the first occuring Entry Point inside EPSM*
- *the second PTSL entry belongs to the second occuring Entry Point inside EPSM*
- *...*
- *the last PTSL entry belongs to the last occuring Entry Point inside EPSM*

Each PTSL entry is defined as

	Contents	Number of Bytes
(1) PTS	*PTS of the corresponding Entry Point*	*4*
	Total	*4*

(1) PTS

PTS of the corresponding Entry Point.

THOMSON multimedia CORPORATE RESEARCH HANNOVER	Page 1 of 3 pages	Date	99/03/09
		Index	EP_SUP-1.DOC
		Name	Marco WINTER

THOMSON multimedia CONFIDENTIAL

PD990019

13

Maximum possible Entry Point support

The entries of the following table shows the maximum possible Entry Point support which is storeable by the described configuration. It's the performable support just **after** a SOB recording. If an entry consists of 2 states, separated by a slash, then the entries describe the following:

- left side of the slash: the state just after recording of a SOB
- right side of the slash: the state after a second off line session (e.g. 1 hour in the night)

STB \ Streamer		simple Streamer, less memory	Streamer is simple, add. memory is avail. just enough for EPs	Streamer is simple, add. memory is avail. more memory	Streamer with dedicated hardw. to parse streams, less memory	Streamer with dedicated hardw. to parse streams, add mem is avail
simple STB	coarse	-	-	-	SOBU	SOBU
	fine	-	-	-	- / packet	packet
	last	-	-	-	- / packet	packet
	PTS	-	-	-	- / yes	yes
	stream	-	-	-	yes	yes
STB sends EP list after record.	coarse	2 SOBU/SOBU	2 SOBU/SOBU	SOBU	SOBU	SOBU
	fine	APAT/packet	APAT/packet	packet	APAT/packet	packet
	last	APAT/packet	APAT/packet	packet	APAT/packet	packet
	PTS	yes	yes	yes	yes	yes
	stream	- / yes	- / yes	- / yes	yes	yes
STB sends EPs during record.	coarse	SOBU	SOBU	SOBU	SOBU	SOBU
	fine	- / packet	packet	packet	- / packet	packet
	last	- / packet	packet	packet	- / packet	packet
	PTS	-	yes	yes	- / yes	yes
	stream	yes	yes	yes	yes	yes

Some Explanations to use the Entry Point Support table:

SOBU: desired application packet is in the indicated SOBU

2 SOBU: desired application packet is in the indicated SOBU or in the following SOBU.

APAT: complete APAT of the desired application packet. The streamer isn't able to calculate directly the sector and appl. packet number from the APAT, i.e. an access to the application must be done via the MAPL.

packet: exact and direct appl. packet location (location is given by a sector number and the application packet number inside this sector).

The various DVD Streamer types are listed horizontally:

- **simple Streamer, less memory:** It's a streamer without any dedicated knowledge about the application (STB). And the streamer has just enough RAM to store a list which indicates the SOBUs containing an EP (coarse list).

14

- **Streamer is simple, add. memory is avail.:** it's similar to the previous described streamer. The only different is
  - **just enough for Eps:** the streamer has additional RAM to store the complete EP information (coarse list + EP start location + EP end location + PTS)
  - **more memory:** the streamer has additional RAM to store the complete EP information (coarse list + EP start location + EP end location + PTS) and the exact packet location + ATS inside the RAM for each incoming application packet during recording
- **Streamer with dedicated hardw. to parse streams, less memory:** the streamer has just enough RAM to store a list which indicates the SOBUs containing an EP. The streamer knows the application, i.e. the streamer is able to find the EPs (start, end and PTS) during recording (and playback) due to the implemented stream parser.
- **Streamer with dedicated hardw. to parse streams, add mem is avail:** this streamer has additional RAM to store the complete EP information (coarse list + EP start location + EP end location + PTS). The streamer knows the application, i.e. the streamer is able to find the EPs (start, end and PTS) during recording (and playback) due to the implemented stream parser.

The various application (application = STB) types are listed vertically:
- **simple STB:** the application doesn't know the existence of streamer.
- **STB sends EP list after record.:** the application knows that a streamer records the sent application packets. After recording of a take (SOB) the application sends a list of EPs (EP start ATS + EP end ATS + PTS) to the streamer.
- **STB sends EPs during record.:** the application knows that a streamer records the sent application packets. During recording of a take (SOB) the application sends parallel (e.g. via an isochronous channel) EPs (EP start ATS + EP end ATS + PTS) to the streamer.

One Entry Point consists of 4 information:
- **coarse:** coarse list. The list describes the SOBUs which have an EP
- **fine:** fine list. This list describes the unambiguous location of the EP either as APAT or as sector number + application number inside this sector
- **last:** fine list of the last application packet which belongs to this EP. It's also a list of the unambiguous location of each EP either as APAT or as sector number + application number inside this sector
- **PTS:** list of PTS'. Each EP has exact one PTS

The point **stream** means EP marks inside the stream. If "yes", then the stream contains additional information for the streamer to detect the application packets which contain an EP start or an EP end.

15

The invention concerns a method and an apparatus for implementing trickplay modes in a data stream recorder.

The following abbreviations are used in the description: DVD: digital versatile disc, LB: logical block, RBN: relative byte number, RBP: relative byte position, RLBN: relative logical block number, STB: set top box, TOC: table of content, SCR: system clock reference, SOB: stream object, DVD RTRW: DVD realtime rewritable, PES: packetised elementary stream, PTS: presentation timestamp, DTS: decoding timestamp, ATS: application timestamp.
Stream recording assumes an application device, e.g. a settop box, connected to a DVD Streamer. Both devices are connected via e.g. an IEEE1394 (IEC 611883) interface including transmitting and receiving firmware.
Fig. 1 shows a simplified block diagram of a settop box AD and a Stream recorder device STRD. AD interacts via an interface IF, e.g. an IEEE1394 interface, with STRD. AD sends its data via output buffering & timestamping handling means BTHOAD to IF and receives from IF data via input buffering & timestamping handling means BTHIAD. A streamer STR within STRD sends its data via output buffering & timestamping handling means BTHO to IF and receives from IF data via input buffering & timestamping handling means BTHI.
Instead of an IEEE1394 connection any other network like the Ethernet or the Internet can be used.

Instead of a settop box any other data stream source can be used, e.g. a DVD player or a PC or Internet receiver. In that case ANT and TU is replaced by e.g. an optical disc and a pickup.

The DVD Stream Recording system is designed to use rewritable DVD discs for recording existing digital bitstreams, editing them and playing them back as bitstreams. This system is designed to satisfy the following requirements:

- Any packet size is supported as long as it is equal or less than 2kByte and is of constant length within a take.
- A timing mechanism, i.e. a time stamp is added to every broadcast packet to enable proper packet delivery during playback.
- To enlarge the fields of applications, non-real-time recording should be possible. However, in this case the STB has to generate the timestamp information.
- Data allocation strategy and a file system to support real-time stream recording.
- Many digital services require Service Information which normally is embedded in the real-time stream. To support a STB fed by data from a DVD player, the DVD should provide additional space, which can be used by the STB to duplicate part of the service information and to add additional TOC information.
- Copy Protection must be supported. In addition, any scrambling performed by the service provider or the STB must be kept unchanged.

User requirements can be grouped into requirements for recording, requirements for playback, and requirements for editing:

Real-time Recording

The system is designed to enable real-time recording of digital streams. It allows the user to concatenate recordings, even if those recordings consist of different stream formats. If recordings are concatenated, a seamless or close-to-seamless playback feature can be achieved, but is not required.

Navigation Support

To support navigation two pieces of information (lists) are generated during recording:

1) An 'original' version of a play list. This list contains quite low level information, e.g. time map or (broadcast)

packet order of the recording. This list is accessible by the STB and the content is understood by the DVD streamer as well as by the STB. In its original version the playlist enables the playback of a complete recording. The playlist may be accessed and extended after recording by the STB to allow more sophisticated playback sequences.

2) The second piece of information, a mapping list, is generated to support the stream recorder to retrieve packet stream chunks (cells), that are described in terms of the application domain, e.g. 'broadcast packets' or 'time'. This list is owned and understood by the DVD streamer only.

Content Description

The system can reserve space which can be used by the STB to store high-level TOC and Service Information. This information is provided for the user to navigate through the content stored on disc and may contain sophisticated EPG information. The content needs not to be understood by the stream recorder. However a common subset of the TOC information, e.g. based on a character string, may be useful to be shared between STB and DVD, in order to enable the stream recorder to provide a basic menu by itself.

Playback of individual recording and playing all recordings sequentially is possible via a play list.

Player menus for entry point selection

The STB can generate a sophisticated menu based on the TOC information stored on the disc. A simple menu is generated by the streamer itself, e.g. via some 'character' information which is shared by STB and DVD.

The DVD streamer creates the 'original version' of the play list. It can allow extensions and modifications of the play list by the STB for more sophisticated playback features. The DVD streamer is not responsible for the content of those sophisticated playlist(s).

The system supports the deletion of single recordings on user's request. Preferably the system allows this feature under the control of the STB.
The system may support insert editing.

Concerning the directory and file structure, the organisation of Stream Data and Navigation Data of DVD Stream Recording is done in a specific way such as to take into account the following:

- Any DVD Streamer device has certain requirements to store its own housekeeping data or Streamer-specific navigation data on the disc. These data are solely for helping the retrieval of recorded data; they need not be understood or even be visible to any outside application device AD.
- Any DVD Streamer device needs to communicate with the application device AD it is connected to. This communication is as universal as possible so that the maximum possible range of applications can be connected to the Streamer. The Navigation Data to support such communication are called Common navigation data and must be understandable by the Streamer as well as by the application device.
- The Streamer device offers to the connected application device AD a means for storing its own private data of any desired kind. The Streamer needs not to understand any of the content, internal structure, or meaning of this application-specific navigation data.

A possible directory and file structure is described below. The files storing the disc content are placed under the STRREC directory which is under the root directory. Under the STRREC directory the following files are created:

- COMMON.IFO
  Basic information to describe the stream content. Needs to be understood by the Application Device as well as the Streamer.

19

- STREAMER.IFO
  Private housekeeping information specific to the Streamer Device. Needs not to be understood by the Application Device.
- APPLICAT.IFO
  Application Private Data, i.e. information that is specific to the Application(s) connected to the Streamer. Needs not to be understood by the Streamer.
- REALTIME.SOB
  Recorded real-time stream data proper.

Note that except for the files described above, the STRREC directory shall not contain any other files or directories.

Stream Data include one or more 'Stream Objects' (SOBs) which each can be stored as a 'Program stream' as described in ISO/IEC 13818-1, Systems.
A SOB can be terminated by a program_end_code. The value of the SCR field in the first pack of each SOB may be non-zero.
A SOB contains the Stream Data packed into a sequence of 'Stream Packs' (S_PCKs). Stream data can be organised as one elementary stream and are carried in PES packets with a stream_id.
In Stream recording, the application performs its own padding so that the pack length adjustment methods of DVD-ROM Video or RTRW need not to be used. In Stream recording it is safe to assume, that the Stream packets will always have the necessary length.

As shown in Fig. 2, a Stream Pack has 2048 bytes and includes a pack header followed by a Stream PES Packet. A system header may be included in those S_PCKs which are the first S_PCK of a SOB. When a system header is included the length of the remaining Stream PES Packet content may be 2010 bytes, and when not included, 2034 bytes. A pack is recorded in one LB.

20

Claims

1. Method for implementing trickplay modes in a data stream recorder, and corresponding recorder, wherein data entry points are added to the bitstream to be recorded and are used when replaying, and wherein said entry points describe several properties like general information, start location list, end location list and presentation time stamp list.

2. Mothod or recorder according to claim 1, wherein said trickplay modes include fast forward, fast reverse, slow motion, single picture step and/or still picture.

1/1

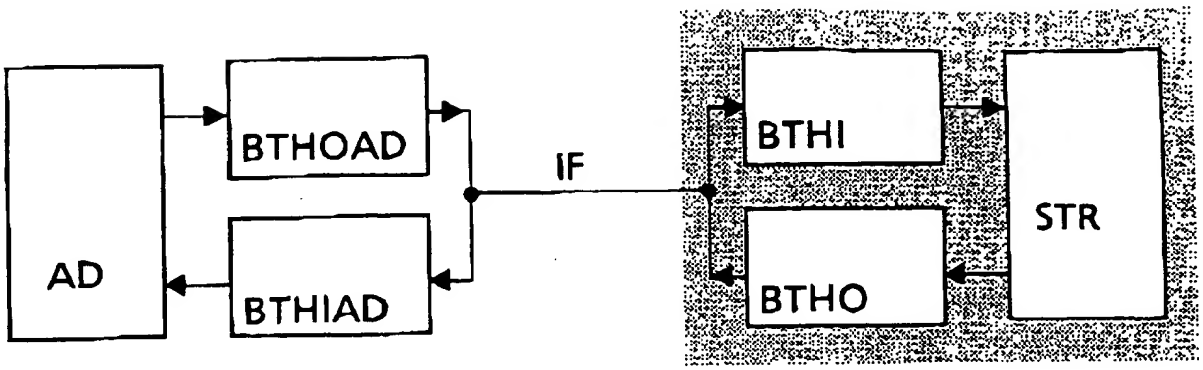

AD
BTHOAD
BTHIAD
IF
BTHI
BTHO
STR
STRD

Fig.1

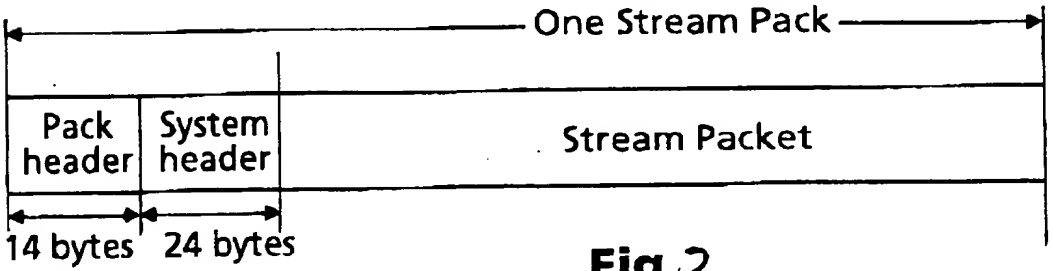

One Stream Pack
Pack header
System header
Stream Packet
14 bytes
24 bytes

Fig.2